# Santa Clara University
## DEPARTMENT of COMPUTER ENGINEERING

Date: June 14, 2007

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

**Clayton Hoefer and Joseph Mastroieni**

ENTITLED

## SCU Stock Market Simulation Model

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF

**BACHELOR OF SCIENCE IN COMPUTER ENGINEERING**

_____
THESIS ADVISOR

_____
DEPARTMENTCHAIR

# SCU STOCK MARKET SIMULATION MODEL

by

Clayton Hoefer and Joseph Mastroieni

SENIOR DESIGN PROJECT REPORT

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Engineering
School of Engineering
Santa Clara University

Santa Clara, California

June 14, 2007

# SCU Stock Market Simulation Model for Classroom Analysis

Clayton Hoefer and Joseph Mastroieni

Department of Computer Engineering
Santa Clara University
2007

## ABSTRACT

Analyzing stock market behavior and learning stock trading techniques can be valuable assets. The financial cost and risk attributed to trading stocks on the stock market often makes this impossible for college students. However, a simulated stock market model would generate the same result without financial risk. In the Santa Clara University (SCU) Stock Market Simulation Model, we created a real-time trading environment that would imitate the New York Stock Exchange (NYSE) model. This project is designed for use in a classroom setting, enabling the professor to manipulate many variables and to launch the simulated stock market on his own time. We had the opportunity through this project to design, develop, and deliver a product to our customer, Professor Sanjiv Das of the Santa Clara University Finance Department. We also experienced new technologies in the effort to best fulfill our customer's requirements. Besides developing a product that would meet the customer's requirements and would function under stress, another vital goal was to design our project for later upgrades. We envision this project being used by the SCU Finance Department in many classes dealing with stock market behavior, and further upgrades will increase the performance and customization of the SCU Stock Market Simulation Model.

**Acknowledgements**

We would like to thank Professor Darren Atkinson and Professor Sanjiv Das for their advice and support throughout the development process of the SCU Stock Market Simulation Model.

# Table of Contents

**Page**

Section 1 – Introduction……...1

1.1 Objectives, Purpose and Goals…….1

1.2 Concept…….....1

1.3 Requirements…….....3

Section 2 – Design Overview……..4

2.1 Introduction……..4

2.2 Development Timeline…….6

2.3 Graphical Layout…….7

2.4 Browser Compatibility…….7

2.5 PHP……..7

2.6 Database Management System……....8

2.7 AJAX……..…11

2.8 Testing…….12

Section 3 – Implementation……...14

3.1 User Interface……..14

3.2 Technical Setup/Prerequisites…….27

Section 4 – User's Manual…….28

4.1 Administrator……..28

4.2 All Traders……..30

Section 5 – Societal Issues…….31

Section 6 – Conclusion……...……...32

Appendix...……...……...……...33

Works Cited……...……36

# List of Figures

Figure 2.1 High Level Technology Diagram……..5

Figure 2.2 Entity Relationship Diagram……..…...9

Figure 2.3 Schema Instance…….…….10

Figure 2.4 AJAX Diagram……11

Figure 2.5 Stock Price Generation (1 of 2) ……..……34

Figure 2.6 Stock Price Generation (2 of 2) ……..…35

## Section 1. Introduction

### 1.1 Objectives, Purpose, and Goals

The objective of our senior design project was to create a web-based stock market simulation model for use by Professor Sanjiv Das in the Santa Clara University Finance department. This simulation model will mimic the NYSE environment by producing realistic real-time information and feedback to traders. At the same time, we can benefit from this project by learning new technologies. We will also get the experience of creating and delivering a fully functional product for our customer (the Finance Department) and "real-world users" (finance students).

The main purpose of creating a stock market simulation model is for stock market analysis. This can be beneficial for finance students who will get the chance to study market trading without risk. It can also benefit those teachers who wish to study stock market behavior by giving them the opportunity to manage a controlled environment. This is possible because the system accepts various administrative parameters, which are fully adjustable.

Our main goal is to provide a fully functional system that meets all the requirements set forward by our customer. These requirements include: real-time information and updates, a real-time stock price graph, adjustable administrator parameters, capability for easy setup to facilitate classroom use and robust code that correctly simulates NYSE rules. Another vital goal was making the website fast, easy to use, and aesthetically pleasing. In addition, our goal was to fully document the project so that later groups could easily understand and make upgrades to our project.

### 1.2 Concept

The concept of the SCU Stock Market Simulation Model is directly linked to the concept of the New York Stock Exchange: to estimate the true price of a stock from freely available information and to act on this information in such a way as to make money. In the real market this information may come in the form of analysis by market experts, news relating to a stock, or

the current bid/ask spread. (The *bid* is the current highest buy order, and the *ask* is the current lowest sell order.)

In our model, traders get information from two sources: news, which is displayed periodically on the trading screen, and the bid/ask spread, which is shown via the stock price graph. In our simulation, just as in the real market, quality of information may vary for different traders. Some traders may get news that accurately predicts the stock's true price, while others have to rely on less accurate information. Note that the news is generated randomly, with the administrator setting the overall accuracy of the news.

To succeed in our simulation, you need an analytical skill that will also serve you well in the stock market: the ability to piece together information on a stock to determine its value, and then act accordingly. Knowing whether to buy or sell a security is the key to making money on stocks, and the only way to gain this knowledge is through utilizing the information available.

### 1.2.1 The Security Specialist

Certain traders, called specialists, are historically hired to make transactions by matching trades on the floor of the New York Stock Exchange. As specialists, these traders are given more information than other traders in order to do their job. This leads to a situation where specific traders possess "inside information": all the prices at which other traders are willing to trade the stock. Whether specialists can effectively use this information to make more intelligent decisions in the simulation is a factor of their analytical skill. Regarding the simulation, the administrator can choose up to one specialist for each stock. Game behavior will remain the same with or without specialists in the game.

## 1.3 Requirements

The Stock Market Simulation Model must meet the following requirements. Many of these requirements have been set by our customer, Professor Das.

### 1.3.1 General Requirements

- The trading game will be Web-based
- The trading interface will be aesthetically pleasing and easy to use
- The trading interface will be fast and responsive to user input
- The software will be designed to be upgradeable. Possible upgrades include the addition of market orders and option and bond trading.
- The software will come with an installer
- The game will be 'moderately' secure. This means an experienced hacker could probably break it, but a general computer user would be unable to tamper with the game.

### 1.3.2 Trading-Specific Requirements

- Traders can submit limit orders to buy or sell securities
- Traders can view their account balance of both securities and cash
- Traders can view their pending orders
- Traders can cancel their pending orders
- Traders can view the bid/ask spread for a given security
- Traders can view a price graph for a given security: this is the graph of transactions involving the given security that have occurred over the course of the game
- Traders designated as 'specialists' can view the limit order book for their security: this includes all of the orders that have been placed for that security
- The game will realize orders automatically when matches are found
- Traders' accounts will be updated automatically when transactions are realized
- The game administrator can set the start and end time of the game
- The game will simulate a trading day: the market will only be open for trading during fixed time periods

- The game will set the ‘real’ price for each security using the stock value tree algorithm provided by Professor Das
- The game will create news events and release this news to a subset of traders
- The administrator can specify the percentage of traders that receive news releases
- The news stories will come from real news sources on the web; the prices will be fictional and generated by the game, as mentioned above
- All data views (prices, transactions, trend) will be updated in real time on traders’ browsers.
- The trader screen will have a ‘ticker tape’ containing stock prices
- The game will have an administrator’s page
- The game will be robust and able to recover from crashers and/or computer failures
- Traders can trade in multiple securities
- Each security will have its own specialist or specialists
- At the end of the game, the winner will be calculated and displayed. Also displayed will be the final standings of all players who participated in the game, sorted by net worth at the end of the game.

## Section 2. Design Overview

### 2.1 Introduction

Our approach to the design process was fairly straightforward. We had clear requirements that had been set down by our customer, and we chose our technology to best fulfill these requirements without compromising the time-frame or scope of the design project. From the requirements, we deduced that the system must be web-based and contain the following pieces: a web server running a server-side scripting language, a relational database for data storage, and client side component to render the user interface and run a client-side scripting language.
We created entity relationship diagrams and a database schema to help with the layout of our relational database. We also built a HTML mock-up to visualize the eventual page layout.

Once we had completed this analysis, the choice of technologies was fairly simple. We decided to build our user interface using XHTML and CSS, which allow complete separation of content and layout declarations, making it easy for us to change the design of the web interface on the fly without affecting the content. We chose Javascript as our client-side scripting language because it is used as a part of a technology called AJAX, which we planned to use to implement the dynamic functionality we wanted in our user interface.

After evaluating a number of server-side scripting languages, we chose PHP for its dynamic capability and for its easy interface with relational databases, namely MySQL. PHP is also open source, well documented and freely accessible. We picked MySQL for our relational database. MySQL is open source and is fully supported in PHP. Finally, Apache was chosen as our web server: it is another highly popular open source program, and also comes installed on Mac OS X, our customer's target platform.

The following is a high level diagram showing how each of above-mentioned technologies fits together:

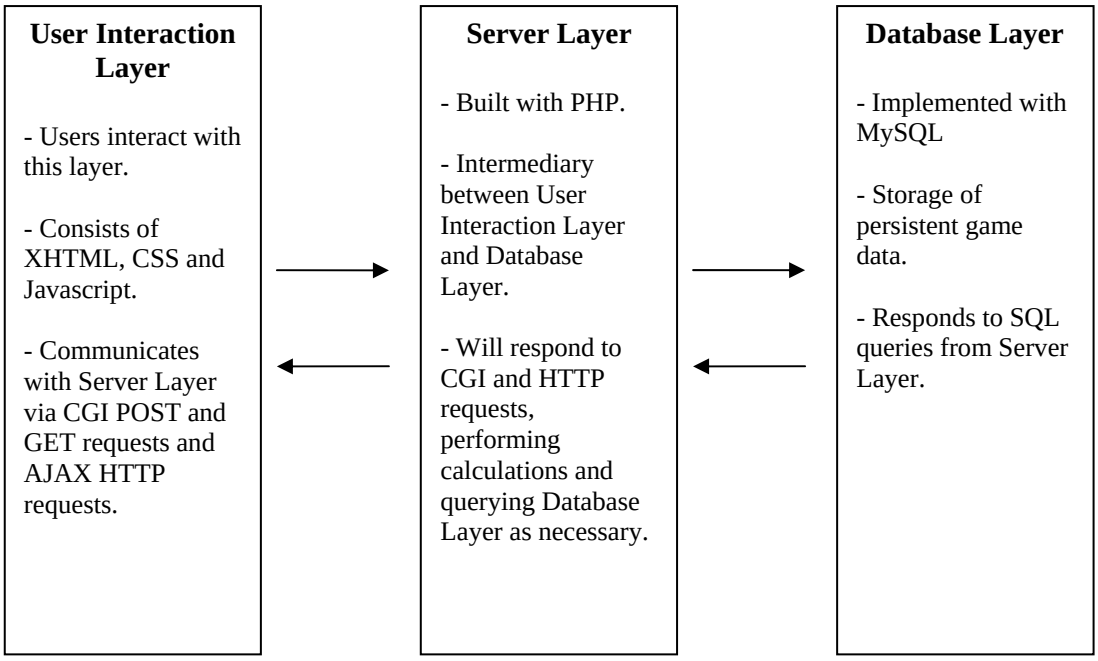

User Interaction Layer
- Users interact with this layer.
- Consists of XHTML, CSS and Javascript.
- Communicates with Server Layer via CGI POST and GET requests and AJAX HTTP requests.
Server Layer
- Built with PHP.
- Intermediary between User Interaction Layer and Database Layer.
- Will respond to CGI and HTTP requests, performing calculations and querying Database Layer as necessary.
Database Layer
- Implemented with MySQL
- Storage of persistent game data.
- Responds to SQL queries from Server Layer.

**Figure 2.1 High Level Technology Diagram**

## 2.2 Development Timeline

The following is a timeline which lays out the high level tasks involved in building the system and a proposed completion date for each. Note that 'Weeks' are in reference to the start of winter quarter 2007.

| Description of Task | Timeframe |
|---|---|
| **Phase One**: Setup file server and SVN repository to allow concurrent development of project from remote locations. | December 2006 |
| Design and implement database in MySQL using the PHP MyAdmin interface. | Weeks 1-2 |
| Build GUI with XHTML and CSS (based on existing HTML mockup). | Weeks 2-3 |
| Add JavaScript client-side input validation to HTML forms in GUI. | Week 4 |
| Implement Phase One* server-side functionality with PHP | Weeks 5-7 |
| Meet with customer to discuss initial operational system | Week 8 |
| **Phase Two**: Enhance PHP functionality and add AJAX features, such as page refresh, to GUI. | Weeks 8-12 |
| Create automated test harness to test PHP functionality and write tests. | Weeks 12-13 |
| Modify code as needed based on test results | Weeks 13-14 |
| Perform acceptance testing with customer and stress testing with 20-30 users | Week 15 |
| Build MAC OS X installer | Weeks 16-18 |
| Make any final changes based on customer requests and stress test results. Retest as necessary. | Weeks 17-20 |

### 2.2.1 Phases of Development

Phase One: The first phase will contain all of the necessary elements required in the bare-bones operational system. The game will have a fully functioning database, limit orders, trading in a single security, no graph and no news releases.

Phase Two: The game will be enhanced by adding the following: trading in multiple securities, graph of each security, and news releases. The GUI will also be enhanced with AJAX features in this phase.

## 2.3 Graphical Layout

We used a combination of Extensible HyperText Markup Language (XHTML) and Cascading Style Sheets (CSS) to implement our user interface. These two technologies are commonly used together to implement professional web pages because of the convenient separation of content and layout declarations that they provide. An XHTML file contains only the content of a web site, along with some simple containers for that content. The CSS then tells the web browser exactly how to render those containers. This allows the programmer to completely change the look and feel of a web page simply by changing the CSS file associated with it. Using XHTML and CSS allowed one person in our group to work on the content of our user interface while the second person separately developed the style of the page.

## 2.4 Browser Compatibility

There are numerous web browsers on the market and we wanted the trading game to be accessible through as many of those as possible. However, due to the limited scope of this project, we decided to limit our browser compatibility to the two most mainstream browsers: Mozilla Firefox 2.0 and Microsoft Internet Explorer 7.

## 2.5 PHP

We considered several options for our server-side scripting language and underlying persistent storage layer before deciding on the combination of PHP and MySQL. PHP was very convenient for this project for several reasons. First, it can be embedded in HTML code, making it easy to see the scripting logic along side the HTML that is rendering its output. Second, it has

an easy to use MySQL API, making interaction with a MySQL database extremely simple. In addition, there are several AJAX libraries written for PHP, which allowed us to generate the dynamic functionality we wanted. Finally, PHP has numerous pre-written graph generation libraries. We used one called *phplot* to generate the bid/ask graph.

### 2.6 Database Management System

There are two main database options for storing information: simple text files or a more sophisticated Database Management System (DBMS). The obvious choice would be XML, or Extensible Markup Language. XML files are highly portable and have some degree of human readability (files can be parsed by a web browser). File locking would be required to prevent simultaneous access of XML files.

There are three major advantages to using a DBMS such as the open source MySQL. First, the data is arranged by a *schema* or structure. Second, there is built-in capacity for *queries* to retrieve data. Third, a DBMS supports *transaction management*, so that concurrent access to data can be made without the introduction of errors in the data. No file locking is needed. Transaction management involves these qualities: Atomicity, Consistency, Isolation, and Durability (ACID). This type of support does not exist for simple text files or XML files. For these reasons, it was decided to use a DBMS (MySQL) for the trading game.

#### 2.6.1 Entity Relationship Diagram

The following diagram represents the tables (entities) and relationships necessary to support the Stock Market Simulation. According to standard practice, a table is represented by a rectangle, a relationship with a diamond, and an attributes as an oval. An attribute which is also a primary key has its text underlined. A relationship that is many-to-one is represented by an arrow pointing towards the entity that contributes one. A one-to-one relationship has an arrow pointing towards each of the two entities.

Note that the table *Portfolios* is a weak entity, that is, it relies not only upon its primary key and also the foreign key of *Traders*.

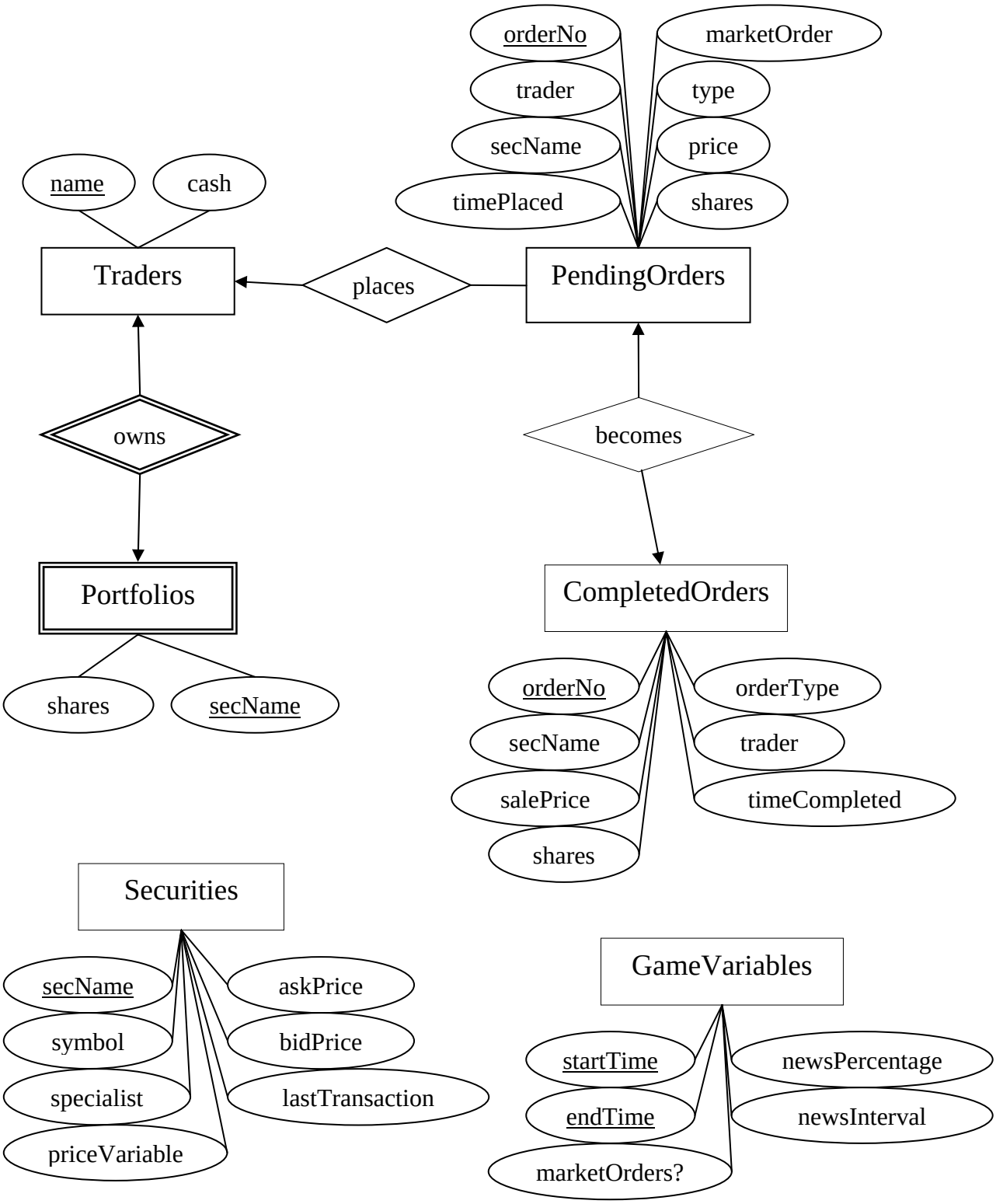



**Figure 2.2 Entity Relationship Diagram**

### 2.6.2 Schema Instance

The following tables are the representation of a particular instance of the schema defined by the E/R diagram above. This schema instance should suggest the proper contents of each attribute, as well as show the relationship between tables.

**Traders**

| *name* | *cash* |
|---|---|
| johnson | 15000 |
| tremain | 10000 |

**Portfolios**

| *(foreign) traderName* | *secName* | *shares* |
|---|---|---|
| tremain | Apple | 500 |
| johnson | Linux | 750 |
| tremain | Microsoft | 250 |

**PendingOrders**

| *orderNo* | *trader* | *secName* | *timePlaced* | *marketOrder* | *type* | *price* | *shares* |
|---|---|---|---|---|---|---|---|
| 1234 | tremain | Apple | 5:10:12 | No | Buy | 15.75 | 20 |
| 1235 | johnson | Linux | 5:12:34 | No | Sell | 17 | 50 |

**CompletedOrders**

| *orderNo* | *secName* | *timeCompleted* | *shares* | *salePrice* | *orderType* | *trader* |
|---|---|---|---|---|---|---|
| 1232 | Apple | 5:05:00 | 500 | 20 | Buy | tremain |
| 1233 | Microsoft | 5:00:04 | 250 | 30 | Buy | tremain |
| 1231 | Apple | 5:05:00 | 500 | 20 | Sell | johnson |
| 1230 | Microsoft | 5:00:04 | 250 | 30 | Sell | johnson |

**Securities**

| *secName* | *symbol* | *specialist* | *priceVariable* | *lastTransaction* | *askPrice* | *bidPrice* |
|---|---|---|---|---|---|---|
| Apple | APPL | tremain | 5 | 20 | - | 15.75 |
| Microsoft | MSFT | johnson | 3 | 30 | - | - |
| Linux | LNUX | johnson | 10 | -_ | 17_ | - |

**GameVariables**

| *startTime* | *endTime* | *newsPercentage* | *newsInterval* | *marketOrders?* |
|---|---|---|---|---|
| 5:00:00 | 6:00:00 | 10 | 25 | Yes |

**Figure 2.3 Schema Instance**

## 2.7 AJAX

Asynchronous Javascript and XML or AJAX is a grouping of technologies commonly used to support dynamic functionality on web pages. It allows a web site to mimic behavior that in the past could only be accomplished on a desktop application. The following diagram shows how the flow of data in an AJAX-based web page compares to a more traditional web site:

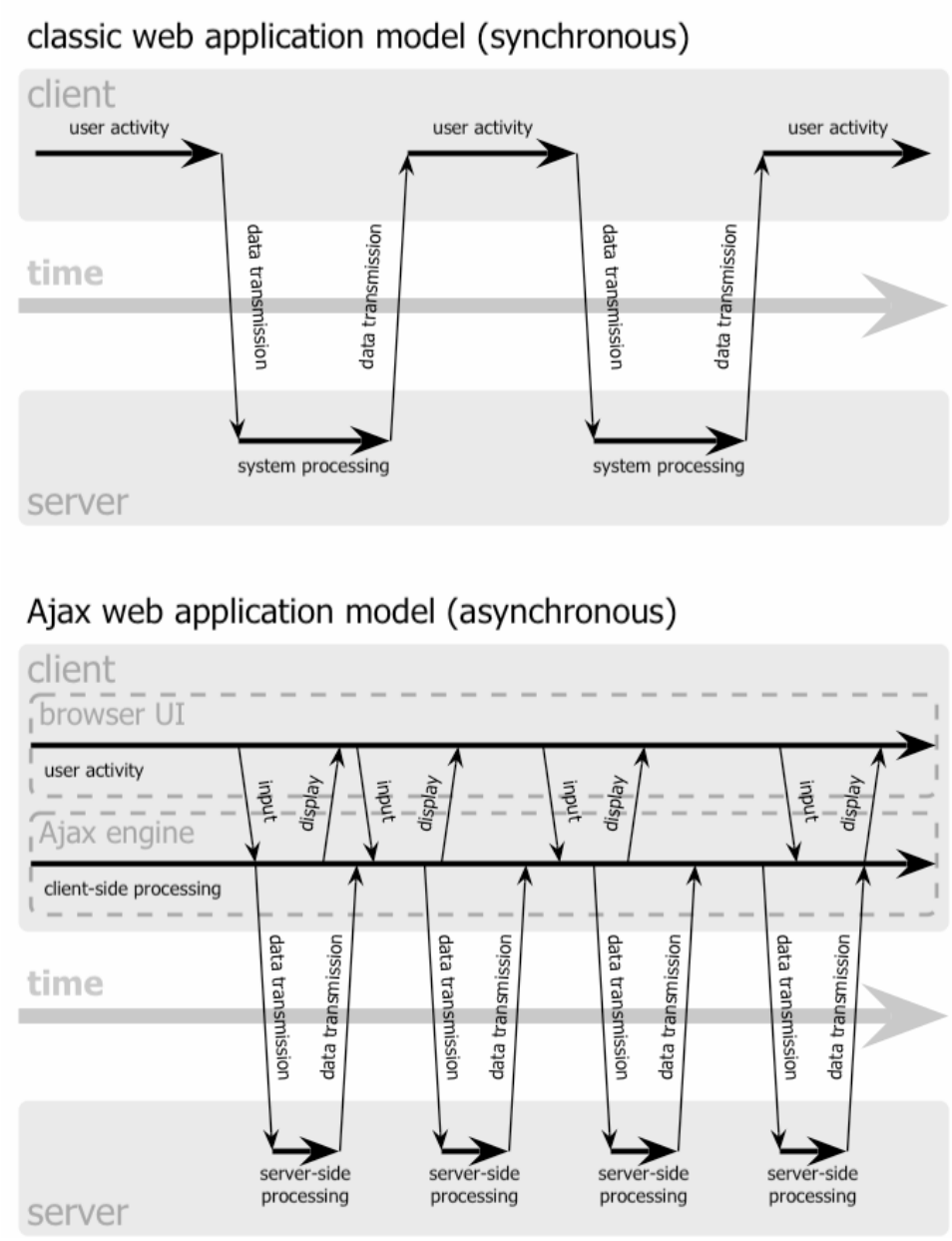



**Figure 2.4 Ajax Diagram**

AJAX allows a client-side scripting language (Javascript in our case) to make calls to functions written in a server-side scripting language (PHP in our case). Javascript also has the capability

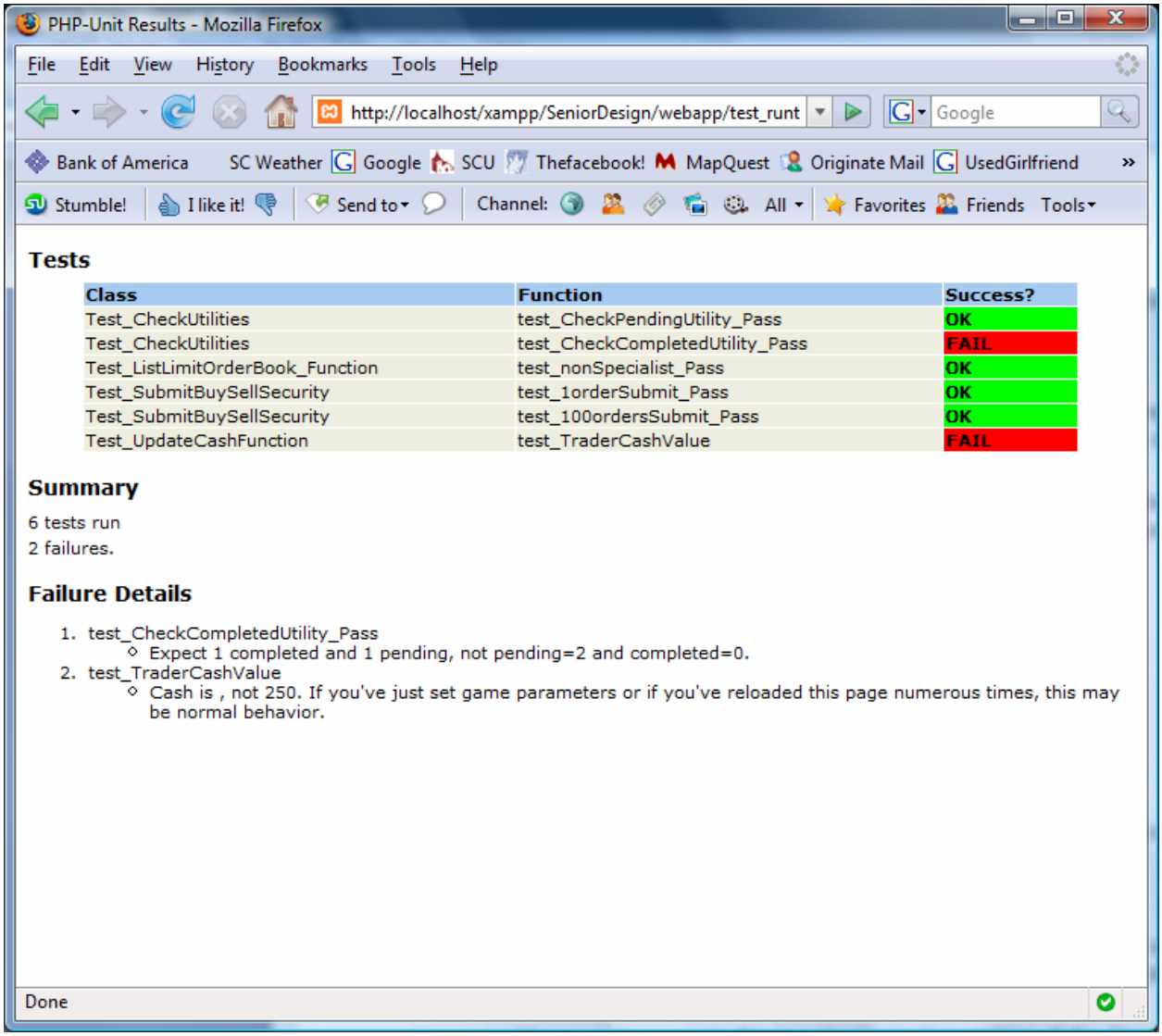
to change individual pieces of a webpage without having to refresh the entire page. With AJAX, Javascript can ask the server-side script to perform some processing, such as querying the database for the latest list of pending orders. The server-side language can then send that information back to the Javascript, which then renders it wherever it chooses on the web page. All that the user sees is a small piece of their web site change, such as their list of pending orders. The rest of the page stays completely static.

We used AJAX to update all of the information on the trading interface: the trader's portfolio, the trader's pending orders, the trader's completed orders, the bid/ask graph of each security and the trader's list of news releases. Since AJAX allows multiple functions to be called concurrently, we could simulate a "real-time" environment by having our Javascript update all of these pieces of the trading screen every few seconds.

### 2.8 Testing

Several methods of testing were used throughout the development process of the trading game. First, we broke down the processing that was required of the PHP into small logical pieces, then wrote a PHP function to perform each of these tasks. Some of these were so small and/or trivial that it was easy to hand-test the function, or simply observe its behavior to ensure that it was performing correctly.

Other functions, such as the one responsible for realizing transactions, were more complex and required an automated testing framework. For this we used a utility called PHPUnit. This utility allows the programmer to write individual tests for a function, then group numerous tests into a test suite, and finally group several test suites together into a complete test run. PHPUnit also has a nicely formatted web-based output that makes it easy to see if tests are passing or failing. The following figure is an example test run output in which you can see that several tests failed. The tool tells us what the expected result was and what the actual result was, allowing us to easily find bugs in our code.


PHP-Unit Results - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/xampp/SeniorDesign/webapp/test_runt
Google
Bank of America SC Weather Google SCU Thefacebook! MapQuest Originate Mail UsedGirlfriend
Stumble! I like it! Send to Channel: All Favorites Friends Tools
Tests
Class Function Success?
Test_CheckUtilities test_CheckPendingUtility_Pass OK
Test_CheckUtilities test_CheckCompletedUtility_Pass FAIL
Test_ListLimitOrderBook_Function test_nonSpecialist_Pass OK
Test_SubmitBuySellSecurity test_1orderSubmit_Pass OK
Test_SubmitBuySellSecurity test_100ordersSubmit_Pass OK
Test_UpdateCashFunction test_TraderCashValue FAIL
Summary
6 tests run
2 failures.
Failure Details
1. test_CheckCompletedUtility_Pass
◇ Expect 1 completed and 1 pending, not pending=2 and completed=0.
2. test_TraderCashValue
◇ Cash is , not 250. If you've just set game parameters or if you've reloaded this page numerous times, this may be normal behavior.
Done

Finally, as the Stock Market Simulation model neared completion, it was important to test how the game would respond under the strain of 20 to 30 users trading simultaneously. While it would have been ideal to implement an automated stress test framework to measure performance, the limited scope of this project did not allow time for this. Instead, we simply used four computers with five to ten browser windows opened on each, all logged in as a different user on the trading game. This was sufficient to show us bottlenecks in our system, most of which were related to PHP, Apache, and MySQL configurations not allowing enough connections. We were able to modify these configurations in order to better optimize the performance of the trading game.

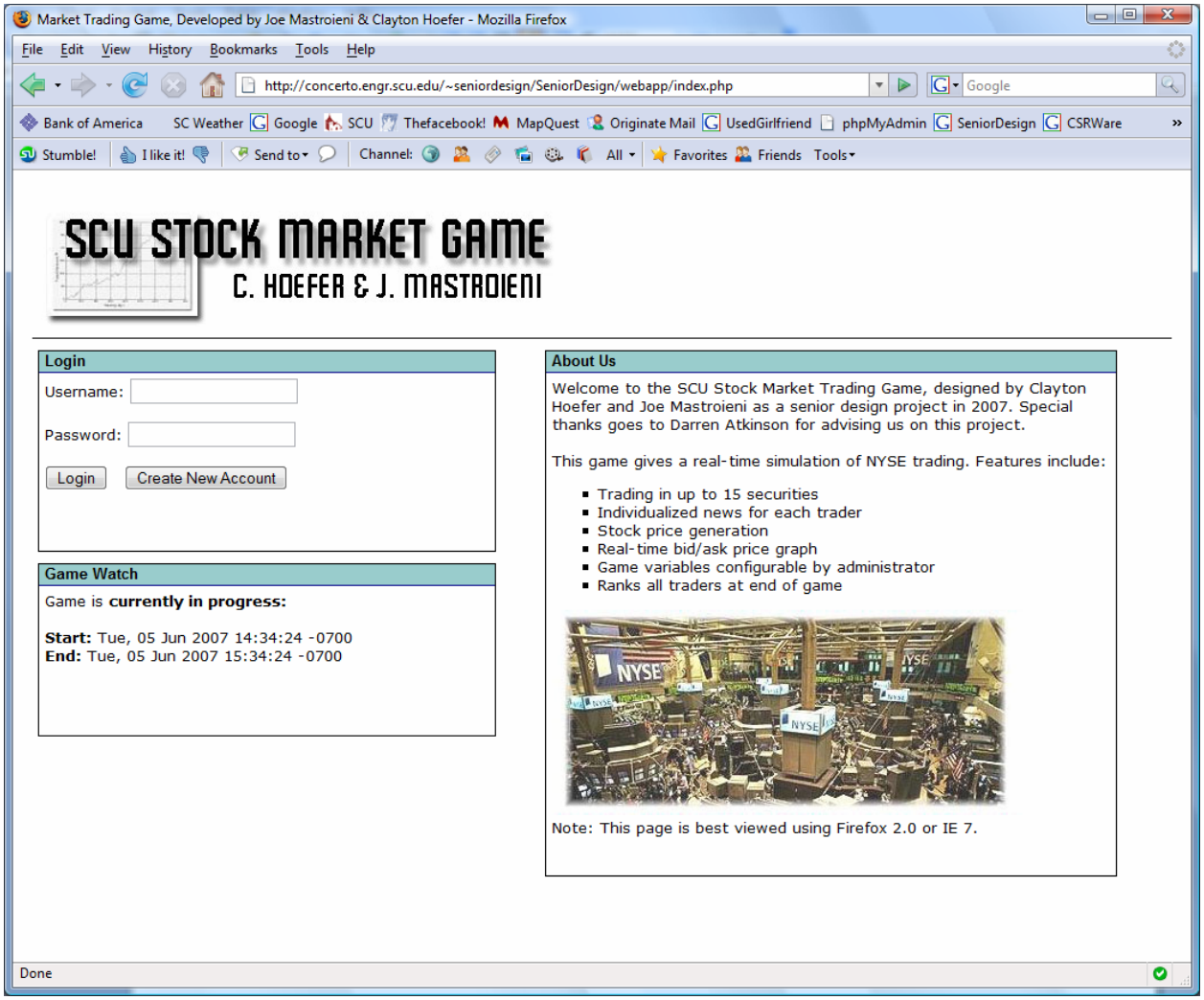
## Section 3. Implementation

### 3.1 User Interface

The following is a look at the Stock Market Simulation Model's user interface. It includes screenshots of all of the screens a typical user (student) would see, as well as the additional screens an administrator (professor) would use to configure the game parameters.

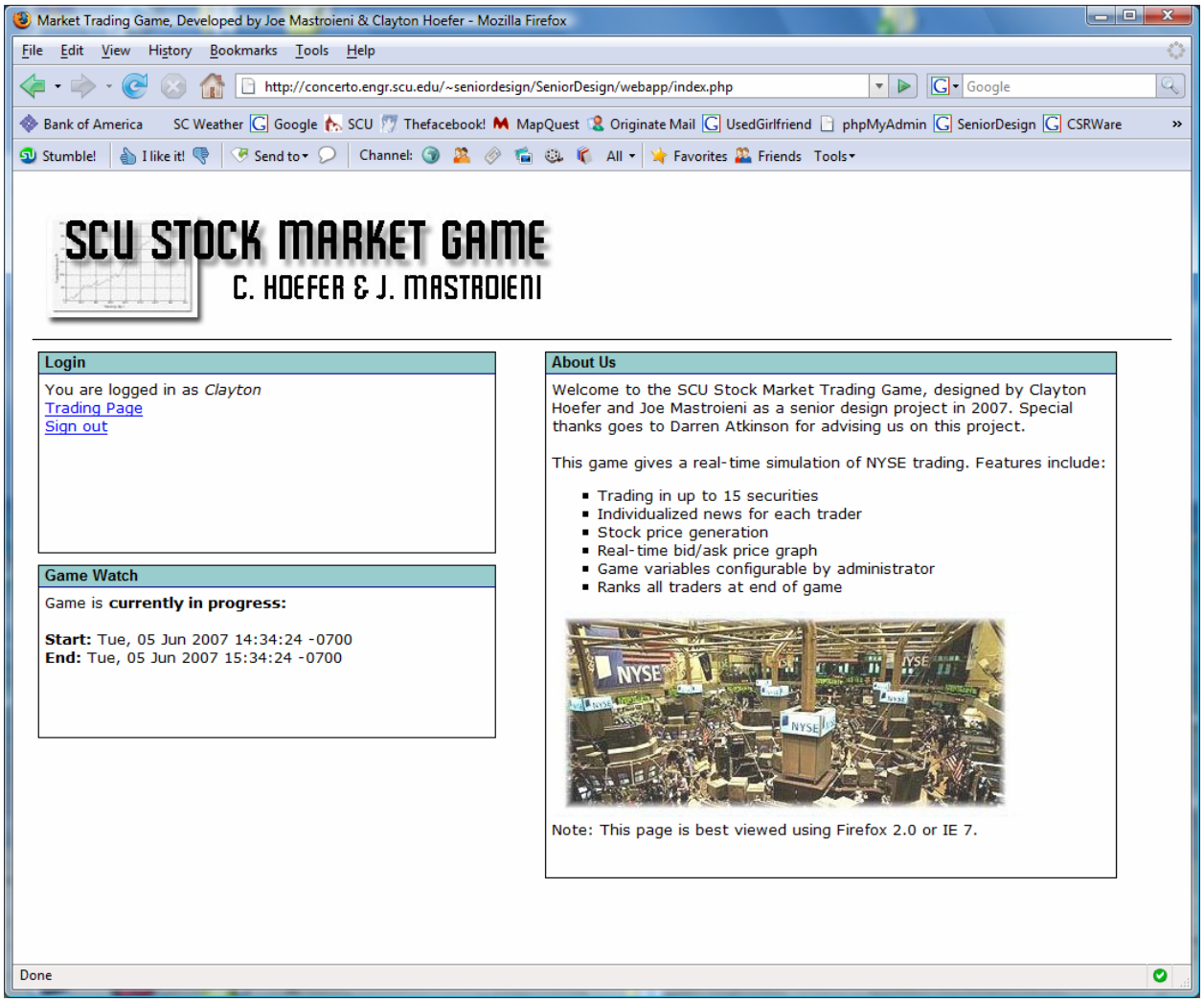
### 3.1.1 Main Page

This is main, or “splash”, page of the trading game. It is the first screen that all users see when navigating to the game web site with their browsers. From there, users can create a user name and password, or log in to the game if they have already done so. The main page also has a box which tells the user that status of the most recently played game (which might still be in-progress). Finally, the main page has a description of the game.


Market Trading Game, Developed by Joe Mastroieni & Clayton Hoefer - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/index.php
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Login
Username:
Password:
Login
Create New Account
Game Watch
Game is currently in progress:
Start: Tue, 05 Jun 2007 14:34:24 -0700
End: Tue, 05 Jun 2007 15:34:24 -0700
About Us
Welcome to the SCU Stock Market Trading Game, designed by Clayton Hoefer and Joe Mastroieni as a senior design project in 2007. Special thanks goes to Darren Atkinson for advising us on this project.
This game gives a real-time simulation of NYSE trading. Features include:
Trading in up to 15 securities
Individualized news for each trader
Stock price generation
Real-time bid/ask price graph
Game variables configurable by administrator
Ranks all traders at end of game
Note: This page is best viewed using Firefox 2.0 or IE 7.
Done

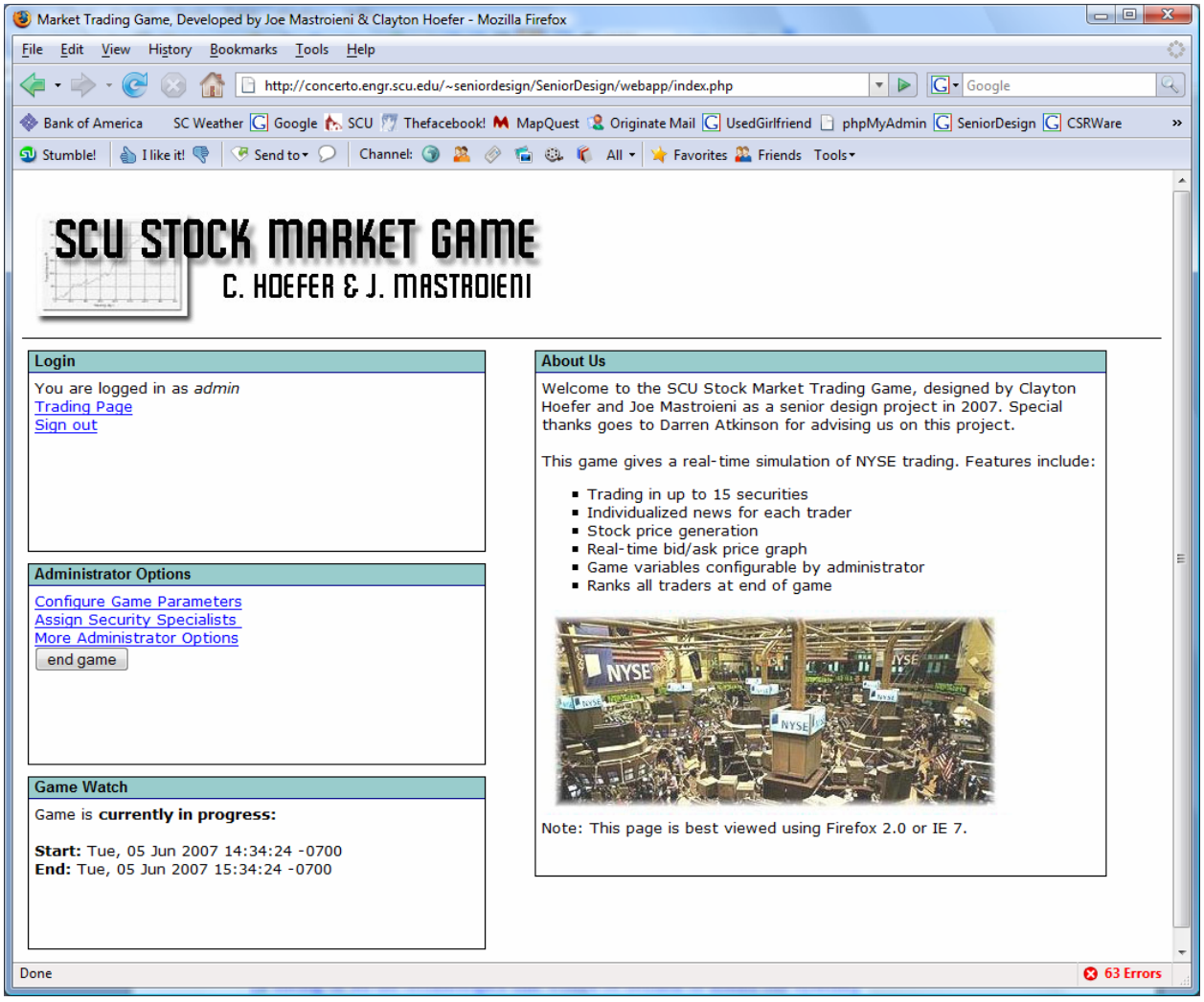
### 3.1.2 Main Page, Logged In as Typical User

The following is a screen shot of the main page once a typical user (student) has logged in. The user now has the option to play the game.


Market Trading Game, Developed by Joe Mastroieni & Clayton Hoefer - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/index.php
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Login
You are logged in as Clayton
Trading Page
Sign out
Game Watch
Game is currently in progress:
Start: Tue, 05 Jun 2007 14:34:24 -0700
End: Tue, 05 Jun 2007 15:34:24 -0700
About Us
Welcome to the SCU Stock Market Trading Game, designed by Clayton Hoefer and Joe Mastroieni as a senior design project in 2007. Special thanks goes to Darren Atkinson for advising us on this project.
This game gives a real-time simulation of NYSE trading. Features include:
Trading in up to 15 securities
Individualized news for each trader
Stock price generation
Real-time bid/ask price graph
Game variables configurable by administrator
Ranks all traders at end of game
Note: This page is best viewed using Firefox 2.0 or IE 7.
Done

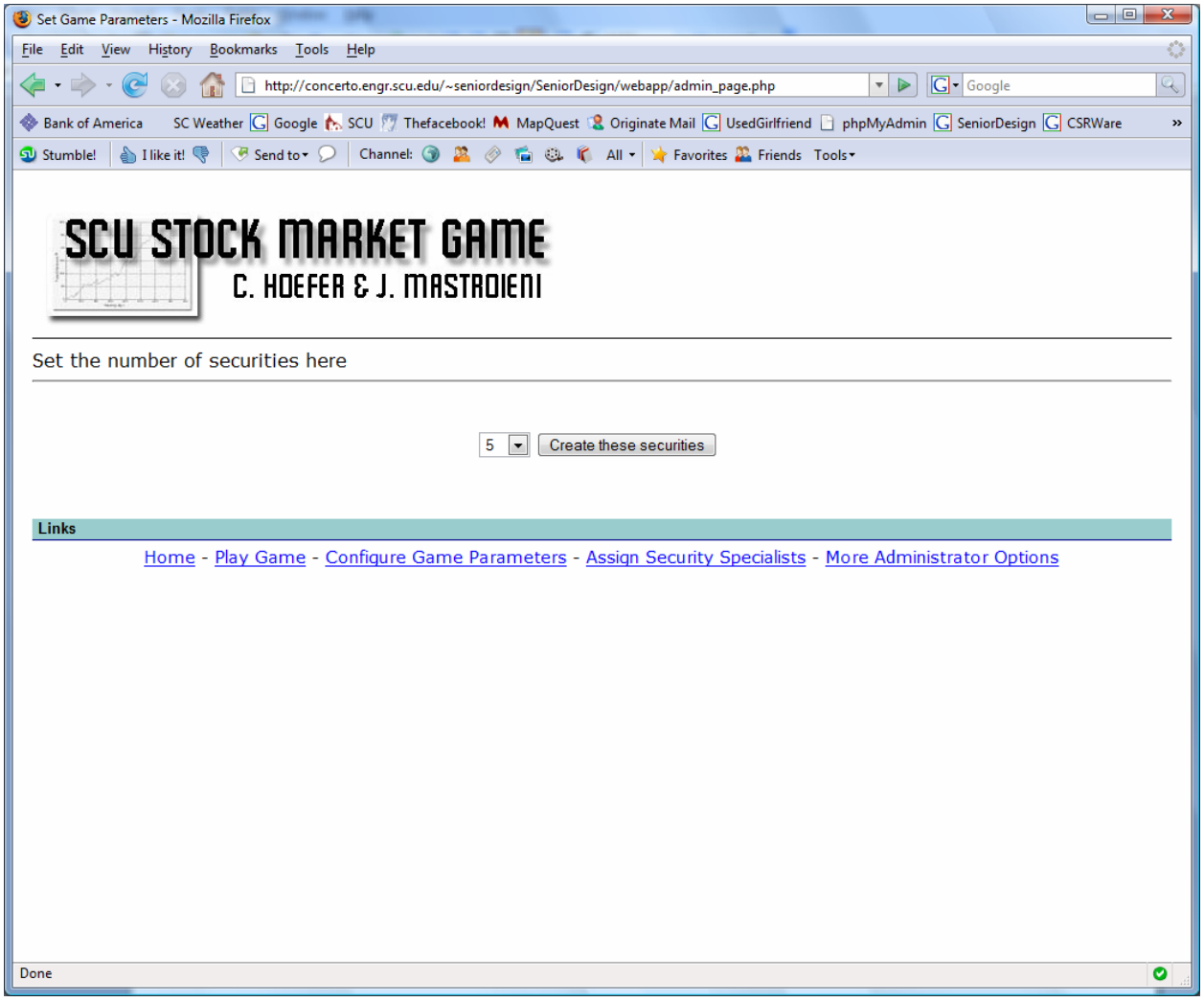
### 3.1.3 Main Page, Logged In as Administrator

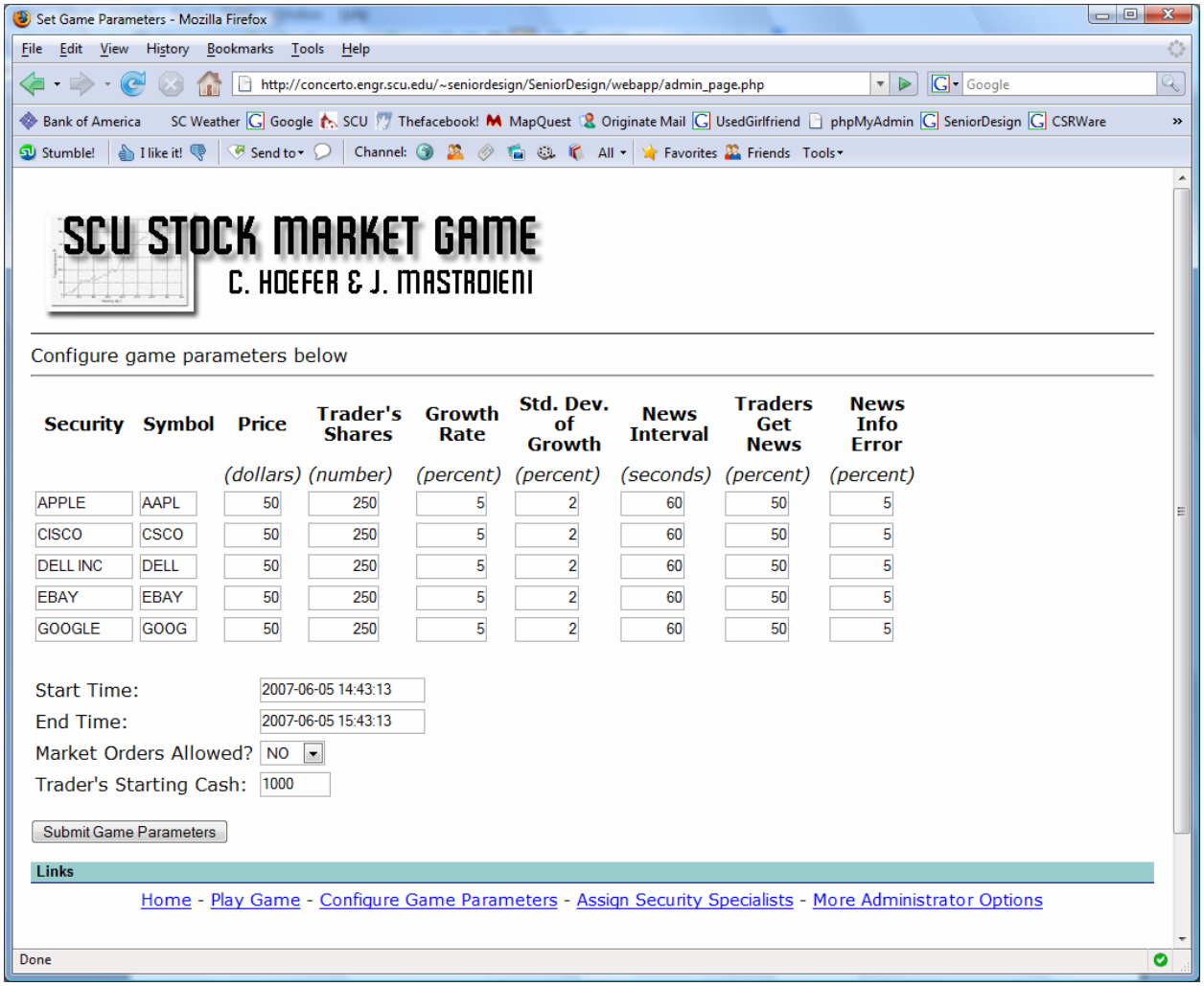
The following shows the main page when the user logs in as the administrator. The user now has a special box containing extra options which allow him or her to configure various game parameters. It also has a button which allows the administrator to instantly end the currently running game.

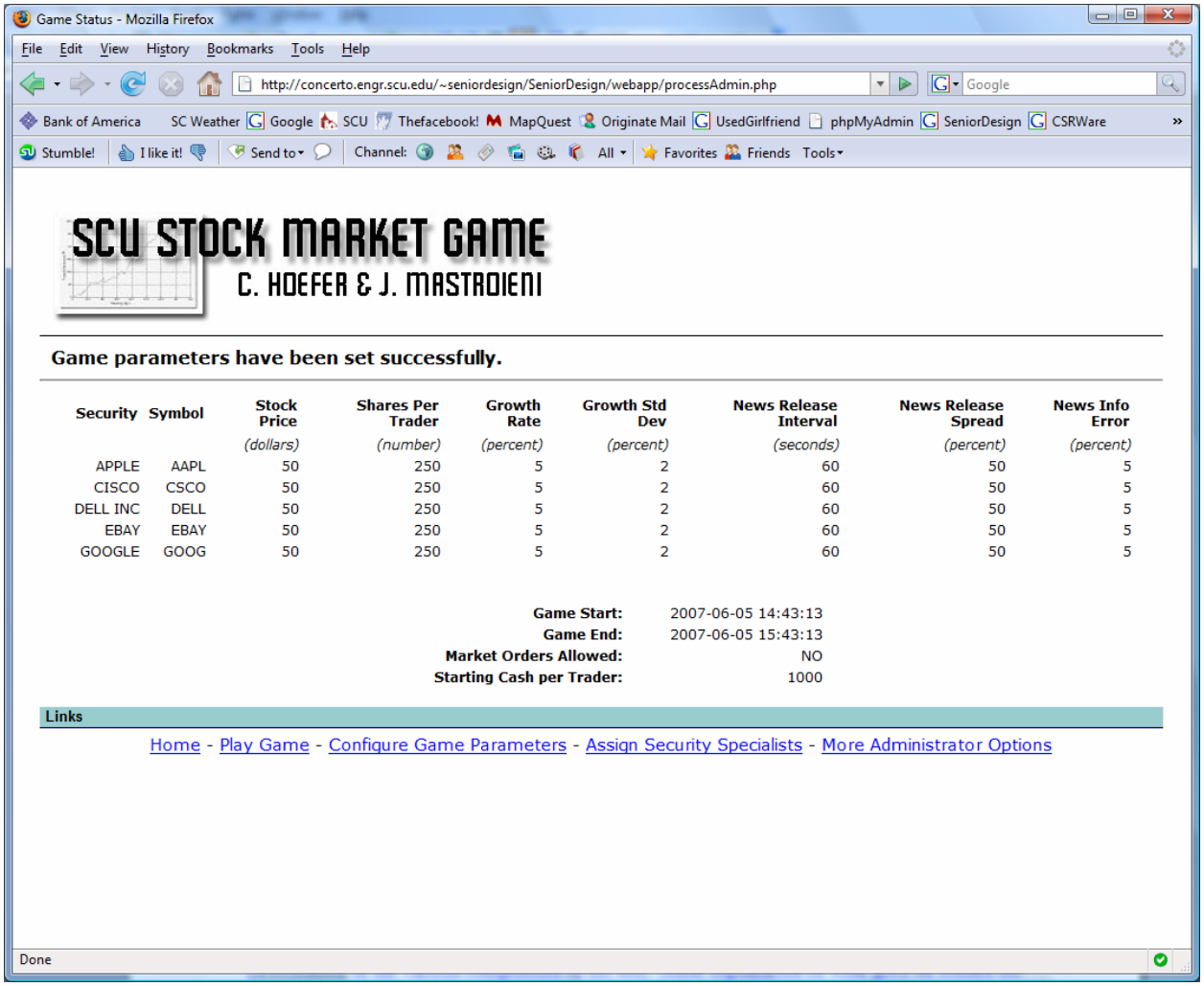

Market Trading Game, Developed by Joe Mastroieni & Clayton Hoefer - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/index.php
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Login
You are logged in as admin
Trading Page
Sign out
Administrator Options
Configure Game Parameters
Assign Security Specialists
More Administrator Options
end game
Game Watch
Game is currently in progress:
Start: Tue, 05 Jun 2007 14:34:24 -0700
End: Tue, 05 Jun 2007 15:34:24 -0700
About Us
Welcome to the SCU Stock Market Trading Game, designed by Clayton Hoefer and Joe Mastroieni as a senior design project in 2007. Special thanks goes to Darren Atkinson for advising us on this project.
This game gives a real-time simulation of NYSE trading. Features include:
Trading in up to 15 securities
Individualized news for each trader
Stock price generation
Real-time bid/ask price graph
Game variables configurable by administrator
Ranks all traders at end of game
Note: This page is best viewed using Firefox 2.0 or IE 7.
Done
63 Errors

### 3.1.4 Administrator Options 1: Configure Game Parameters

The following screenshot shows the first administrative page which allows the administrator to configure the game parameters. This page has three parts: first the administrator has to choose the number of securities that he or she wishes to have in the game. Next the administrator sets the various game parameters associated with each security. Finally, a confirmation page is displayed showing the submitted game parameters.

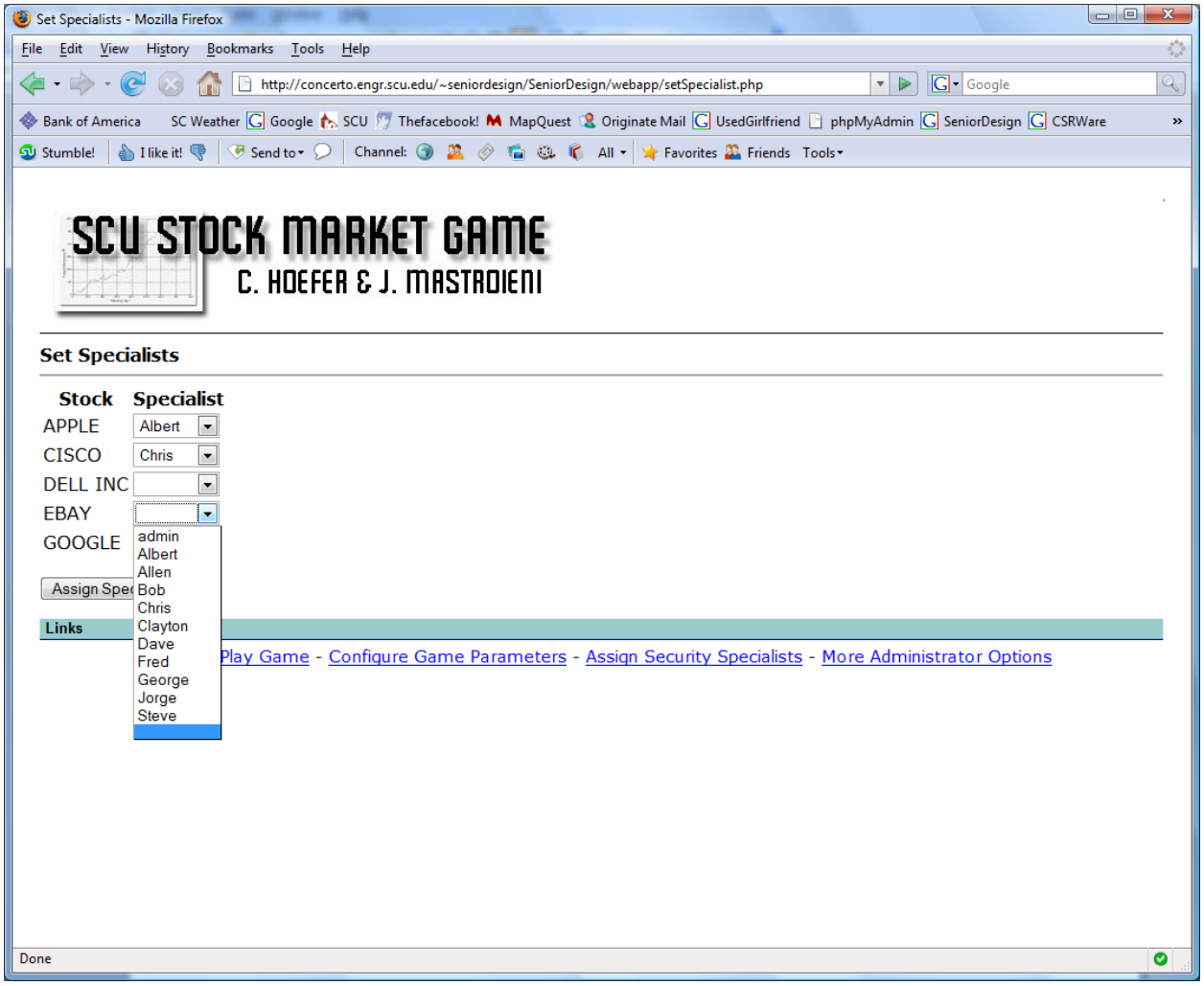

Set Game Parameters - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/admin_page.php
Google
Bank of America SC Weather Google SCU Thefacebook! MapQuest Originate Mail UsedGirlfriend phpMyAdmin SeniorDesign CSRWare
Stumble! I like it! Send to Channel: All Favorites Friends Tools
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Set the number of securities here
5
Create these securities
Links
Home - Play Game - Configure Game Parameters - Assign Security Specialists - More Administrator Options
Done

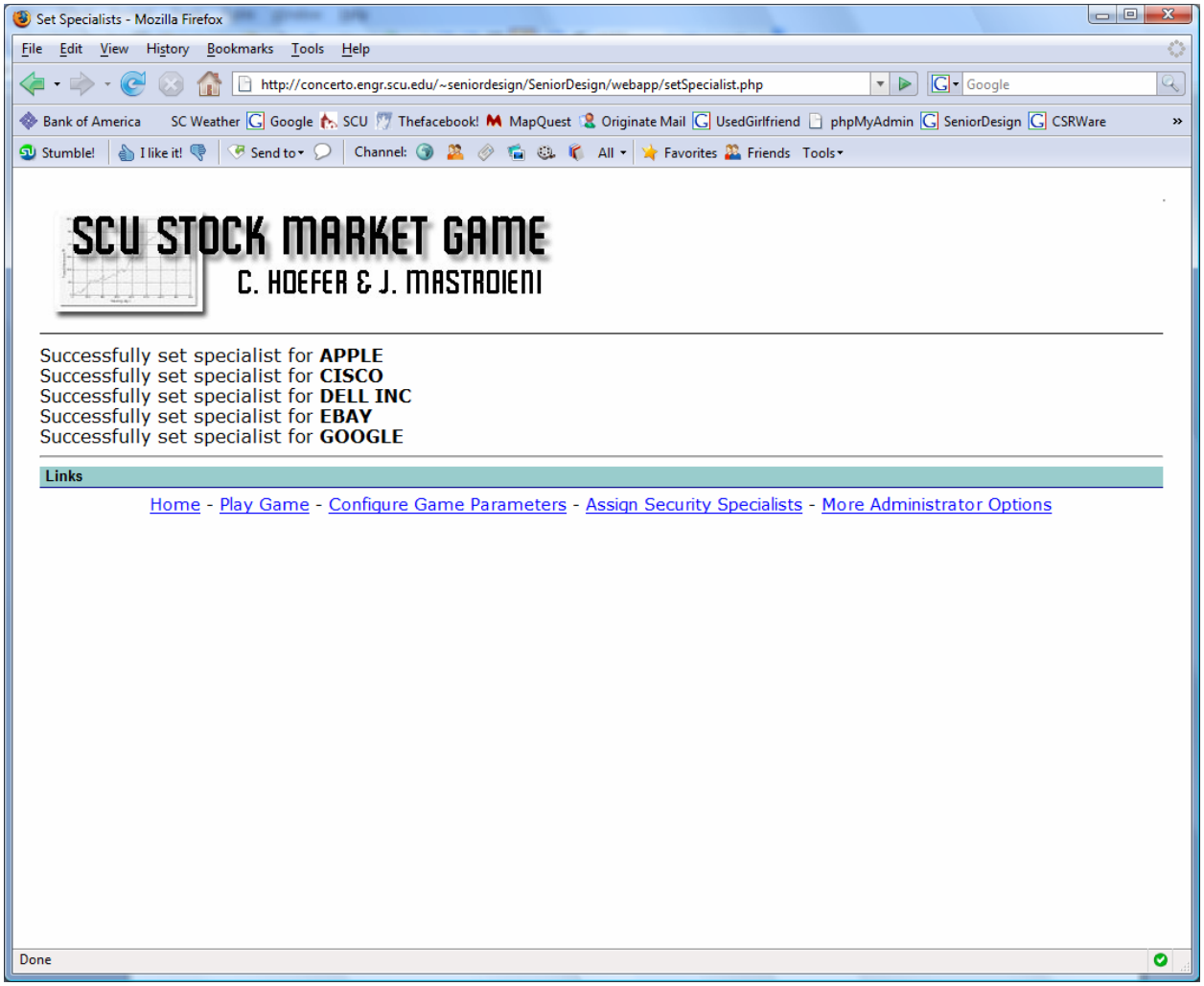

Set Game Parameters - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/admin_page.php
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Configure game parameters below
Security
Symbol
Price
Trader's Shares
Growth Rate
Std. Dev. of Growth
News Interval
Traders Get News
News Info Error
(dollars)
(number)
(percent)
(percent)
(seconds)
(percent)
(percent)
APPLE AAPL 50 250 5 2 60 50 5
CISCO CSCO 50 250 5 2 60 50 5
DELL INC DELL 50 250 5 2 60 50 5
EBAY EBAY 50 250 5 2 60 50 5
GOOGLE GOOG 50 250 5 2 60 50 5
Start Time: 2007-06-05 14:43:13
End Time: 2007-06-05 15:43:13
Market Orders Allowed? NO
Trader's Starting Cash: 1000
Submit Game Parameters
Links
Home - Play Game - Configure Game Parameters - Assign Security Specialists - More Administrator Options
Done


Game Status - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/processAdmin.php
Google
Bank of America SC Weather Google SCU Thefacebook! MapQuest Originate Mail UsedGirlfriend phpMyAdmin SeniorDesign CSRWare
Stumble! I like it! Send to Channel: All Favorites Friends Tools
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Game parameters have been set successfully.
Security Symbol Stock Price Shares Per Trader Growth Rate Growth Std Dev News Release Interval News Release Spread News Info Error
(dollars) (number) (percent) (percent) (seconds) (percent) (percent)
APPLE AAPL 50 250 5 2 60 50 5
CISCO CSCO 50 250 5 2 60 50 5
DELL INC DELL 50 250 5 2 60 50 5
EBAY EBAY 50 250 5 2 60 50 5
GOOGLE GOOG 50 250 5 2 60 50 5
Game Start: 2007-06-05 14:43:13
Game End: 2007-06-05 15:43:13
Market Orders Allowed: NO
Starting Cash per Trader: 1000
Links
Home - Play Game - Configure Game Parameters - Assign Security Specialists - More Administrator Options
Done

### 3.1.5 Administrator Options 2: Assign Security Specialists

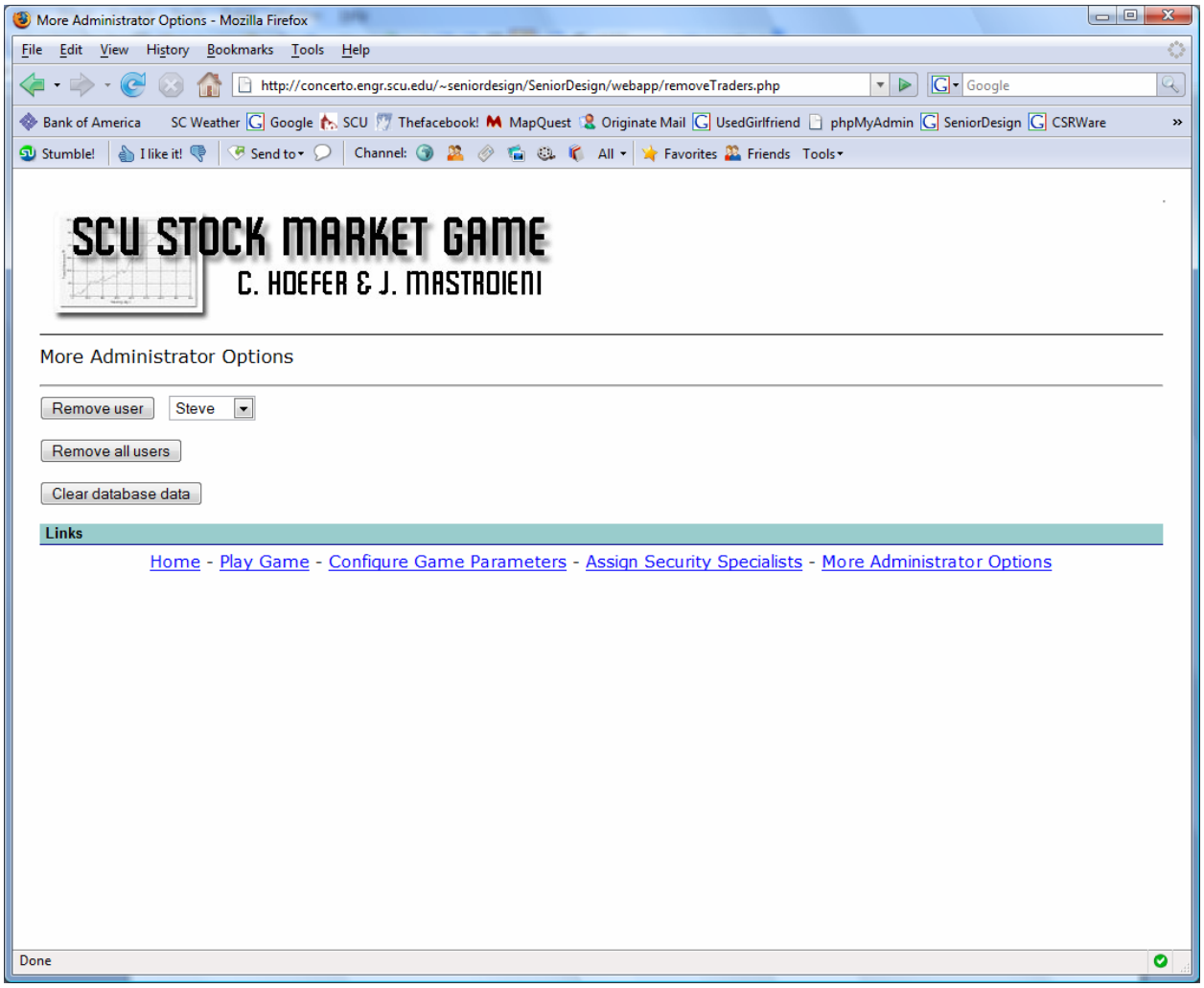
As mentioned above, a security specialist is allowed special privileges. He or she gets to see all of the pending trades for his or her security. The administrator has to assign security specialists from the list of all registered traders. Shown below is the page which allows the administrator to do this and the subsequent confirmation page which appears once the choices have been submitted.




Set Specialists - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/setSpecialist.php
Google
Bank of America SC Weather Google SCU Thefacebook! MapQuest Originate Mail UsedGirlfriend phpMyAdmin SeniorDesign CSRWare
Stumble! I like it! Send to Channel: All Favorites Friends Tools
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Successfully set specialist for APPLE
Successfully set specialist for CISCO
Successfully set specialist for DELL INC
Successfully set specialist for EBAY
Successfully set specialist for GOOGLE
Links
Home - Play Game - Configure Game Parameters - Assign Security Specialists - More Administrator Options
Done

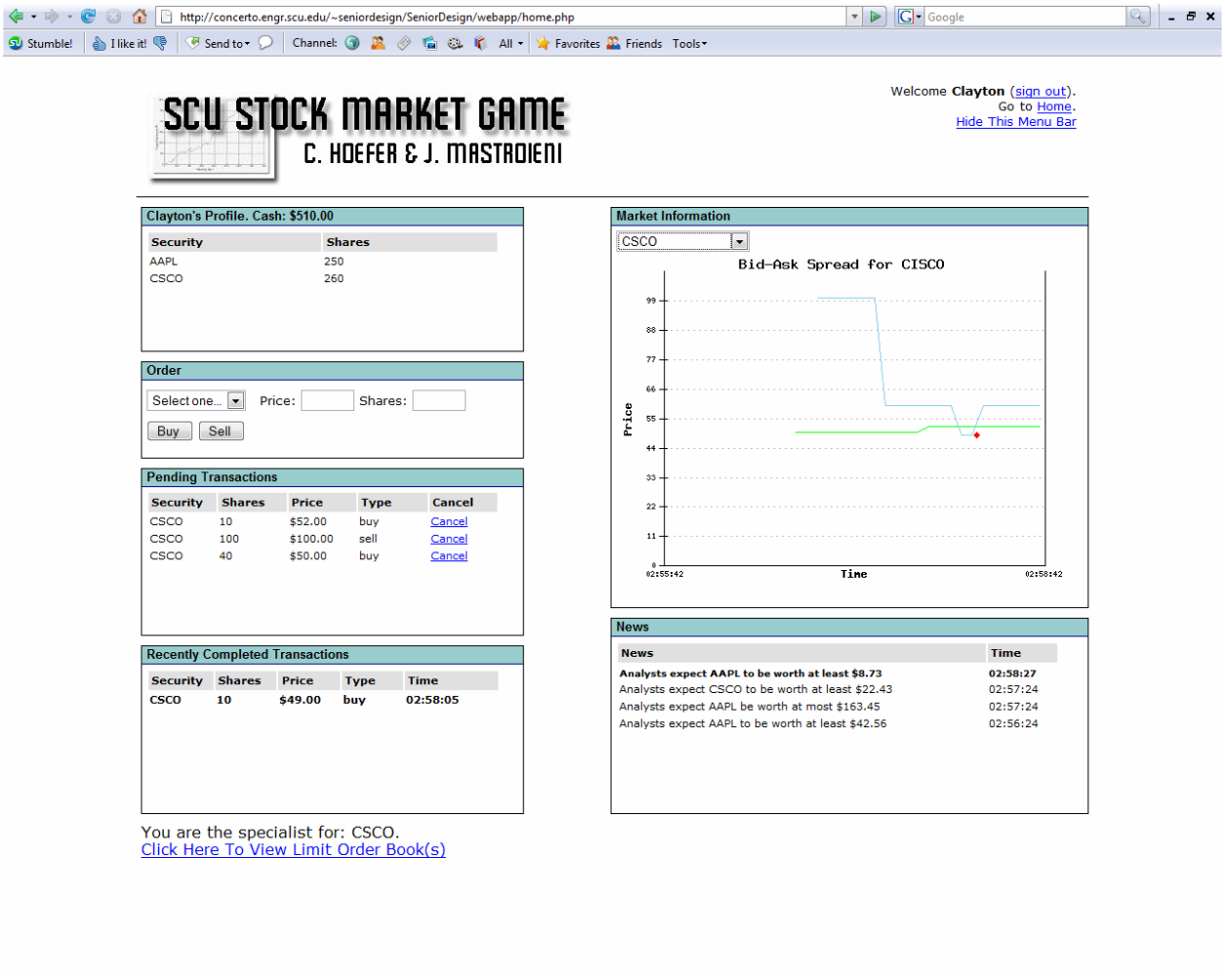
### 3.1.6 Administrator Options 3: More Administrator Options

The following is the last administrative page. It allows the administrator to remove individual users, remove all users or simply clear all data from the database if he or she wishes to start a game entirely from scratch. Each has a simple confirmation page (not shown) which tells if the operation was performed successfully.


More Administrator Options - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/removeTraders.php
Google
Bank of America SC Weather Google SCU Thefacebook! MapQuest Originate Mail UsedGirlfriend phpMyAdmin SeniorDesign CSRWare
Stumble! I like it! Send to Channel: All Favorites Friends Tools
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
More Administrator Options
Remove user
Steve
Remove all users
Clear database data
Links
Home - Play Game - Configure Game Parameters - Assign Security Specialists - More Administrator Options
Done

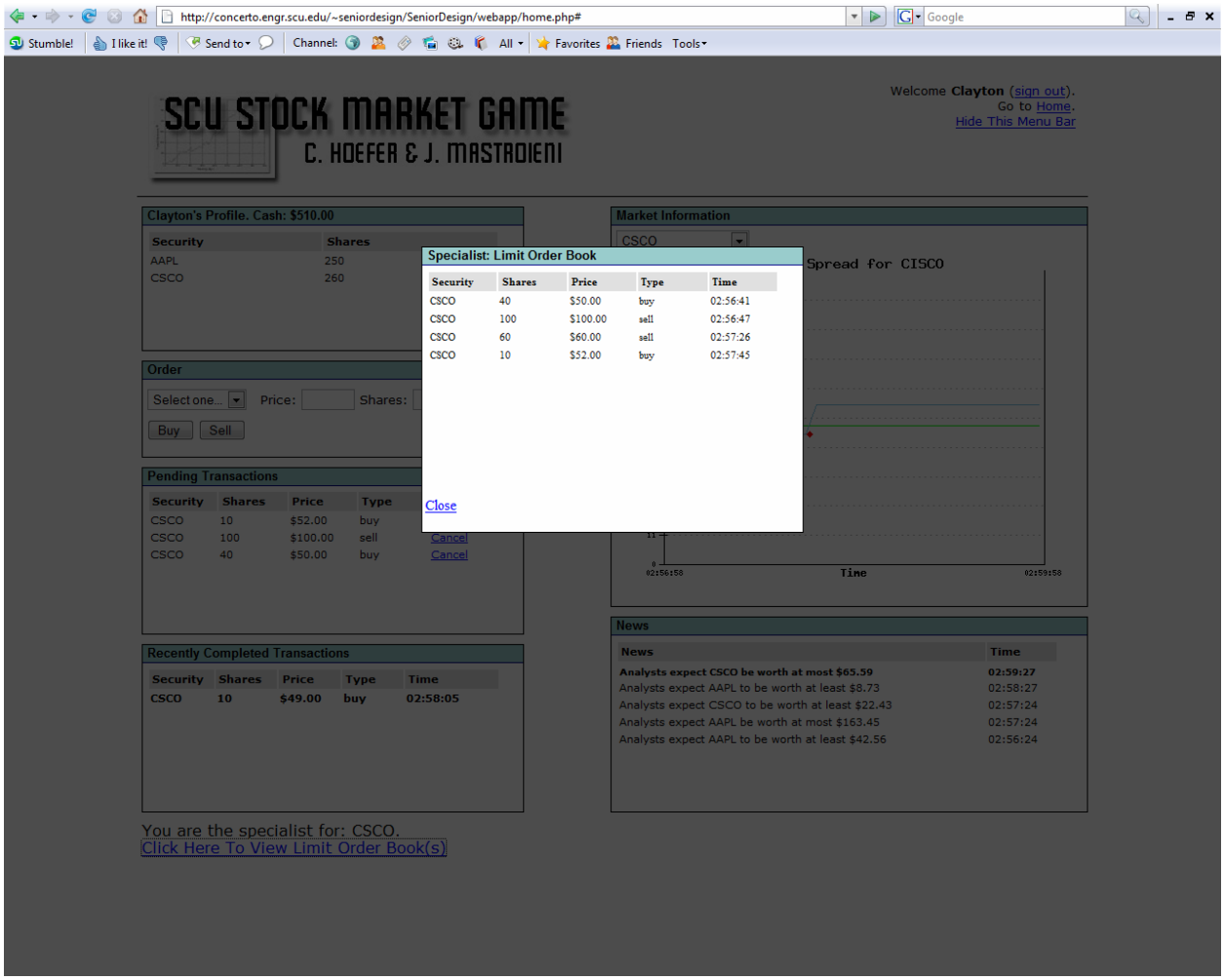
### 3.1.7 Trading Screen

The following is the trading screen that each trader (including the admin) sees. Note that the user who is viewing this screen has been assigned to be the specialist for Cisco and has a special link at the bottom of his screen because of this.


http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/home.php
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Welcome Clayton (sign out).
Go to Home.
Hide This Menu Bar
Clayton's Profile. Cash: $510.00
Security Shares
AAPL 250
CSCO 260
Order
Select one... Price: Shares:
Buy Sell
Pending Transactions
Security Shares Price Type Cancel
CSCO 10 $52.00 buy Cancel
CSCO 100 $100.00 sell Cancel
CSCO 40 $50.00 buy Cancel
Recently Completed Transactions
Security Shares Price Type Time
CSCO 10 $49.00 buy 02:58:05
Market Information
CSCO
Bid-Ask Spread for CISCO
Price
Time
News
News Time
Analysts expect AAPL to be worth at least $8.73 02:58:27
Analysts expect CSCO to be worth at least $22.43 02:57:24
Analysts expect AAPL be worth at most $163.45 02:57:24
Analysts expect AAPL to be worth at least $42.56 02:56:24
You are the specialist for: CSCO.
Click Here To View Limit Order Book(s)

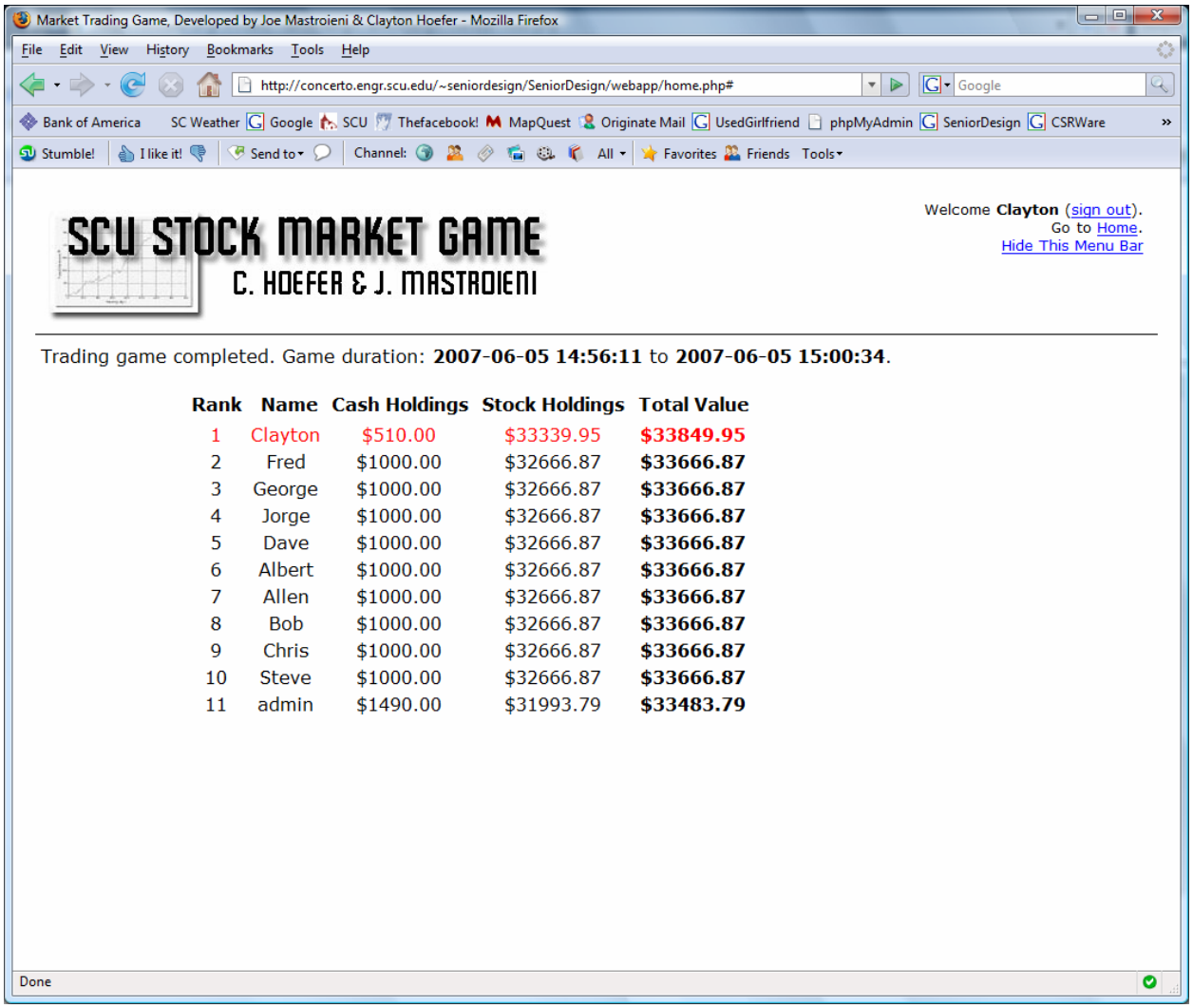
### 3.1.8 Trading Screen: Specialist Window

Since the trader who is currently logged in is a specialist for Cisco, he can view Cisco's limit order book. This book shows all the pending transactions for Cisco. A screenshot of this is shown below.


http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/home.php#
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Welcome Clayton (sign out).
Go to Home.
Hide This Menu Bar
Clayton's Profile. Cash: $510.00
Security Shares
AAPL 250
CSCO 260
Market Information
CSCO
Spread for CISCO
Specialist: Limit Order Book
Security Shares Price Type Time
CSCO 40 $50.00 buy 02:56:41
CSCO 100 $100.00 sell 02:56:47
CSCO 60 $60.00 sell 02:57:26
CSCO 10 $52.00 buy 02:57:45
Close
Order
Select one... Price: Shares:
Buy Sell
Pending Transactions
Security Shares Price Type
CSCO 10 $52.00 buy
CSCO 100 $100.00 sell Cancel
CSCO 40 $50.00 buy Cancel
Time
News
News Time
Analysts expect CSCO be worth at most $65.59 02:59:27
Analysts expect AAPL to be worth at least $8.73 02:58:27
Analysts expect CSCO to be worth at least $22.43 02:57:24
Analysts expect AAPL be worth at most $163.45 02:57:24
Analysts expect AAPL to be worth at least $42.56 02:56:24
Recently Completed Transactions
Security Shares Price Type Time
CSCO 10 $49.00 buy 02:58:05
You are the specialist for: CSCO.
Click Here To View Limit Order Book(s)

### 3.1.9 End of Game

Traders can only trade during the specified game "window", or "trading day". When the game ends, all of the traders in the system are sorted by net worth and the results are displayed in place of the regular trading screen. An example of this is shown below – the trader with the highest net worth is shown in red at the top of the list.


Market Trading Game, Developed by Joe Mastroieni & Clayton Hoefer - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://concerto.engr.scu.edu/~seniordesign/SeniorDesign/webapp/home.php#
Bank of America SC Weather Google SCU Thefacebook! MapQuest Originate Mail UsedGirlfriend phpMyAdmin SeniorDesign CSRWare
Stumble! I like it! Send to Channel: All Favorites Friends Tools
SCU STOCK MARKET GAME
C. HOEFER & J. MASTROIENI
Welcome Clayton (sign out).
Go to Home.
Hide This Menu Bar
Trading game completed. Game duration: 2007-06-05 14:56:11 to 2007-06-05 15:00:34.
Rank Name Cash Holdings Stock Holdings Total Value
1 Clayton $510.00 $33339.95 $33849.95
2 Fred $1000.00 $32666.87 $33666.87
3 George $1000.00 $32666.87 $33666.87
4 Jorge $1000.00 $32666.87 $33666.87
5 Dave $1000.00 $32666.87 $33666.87
6 Albert $1000.00 $32666.87 $33666.87
7 Allen $1000.00 $32666.87 $33666.87
8 Bob $1000.00 $32666.87 $33666.87
9 Chris $1000.00 $32666.87 $33666.87
10 Steve $1000.00 $32666.87 $33666.87
11 admin $1490.00 $31993.79 $33483.79
Done

### 3.2 Technical Setup/Prerequisites

In order to setup the Stock Market Simulation Model, the administrator must have a web server running the following software:

1. Apache with support for PHP5
2. PHP5
3. MySQL

In addition, PHP5 must have the following libraries installed:

1. MDB2: usually installed with the following commands:
```
pear upgrade mdb2
pear install MDB2_Driver_mysql
```

2. APC Cache: usually installed with the following command:
```
pecl install apc
```

The following lines should then be added to the PHP5 configuration (php.ini) file:

```
apc.enabled = 1
apc.shm_segments = 1
apc.shm_size = 128
apc.optimization = 0
apc.num_files_hint = 1000
apc.ttl = 0
apc.gc_ttl = 3600
apc.cache_by_default = On
apc.slam_defense = 0
apc.file_update_protection = 2
apc.enable_cli = 0
apc.stat=1
```

Then, the administrator should install the database using the script in the ‘db’ folder of the trading game:

```
cd <path to game root dir>/db
mysql –u root
source seniordesign.sql;
```

Next, the ‘webapp’ directory of the trading game should be copied to the web server’s public directory (‘Sites’ on OSX).

Finally, the ‘config.xml’ file in the ‘webapp’ directory should be edited with the appropriate parameters for the particular web server. An example configuration is shown below:

```
<config>
	<dbHost>localhost</dbHost>
	<dbName>seniordesign</dbName>
	<dbUserName>sd</dbUserName>
	<dbPassword>pwd</dbPassword>
	<phpPath>/usr/local/php5/bin/php</phpPath>
</config>
```

## Section 4. User’s Manual

### 4.1 Administrator

#### 4.1.1 First Time Setup

1. Follow the directions above to install the Stock Market Simulation Model on your web server.
2. Navigate your browser to the index page.
3. Create a user with the name “admin” and a password of your choosing. This user will be the only administrator for the game.
4. You now have an extra box on the main page called “Administrator Options”.

#### 4.1.2 Configure Game Parameters

The following steps must be followed before each run of the game:

1. Login as “admin”, if not already.
2. Click the “Configure Game Parameters” link in the “Administrator Options” box on the index page.
3. Decide on the number of securities you want to have in the game and click “Create these securities”.
4. Now, specify the various game parameters you would like each security to have. Default values are provided, but you are free to change any of them that you wish.
5. Set the start time, end time, and traders’ starting cash.

6. Click “Submit Game Parameters”.
7. The game is now started. Review the parameters. If any of them are incorrect, restart the game by starting over at Step 1 in this section.

### 4.1.3 Assign Security Specialists

The following steps can be followed before each run of the game (if not followed, no specialist(s) will be assigned for that game).

1. Login as “admin”, if not already.
2. Click the “Assign Security Specialists” link either at the bottom of the confirmation page from the last section or in the “Administrator Options” box on the index page.
3. Choose a user from the list of registered traders to be a specialist for each security you want to assign a specialist to.
4. Click “Assign Specialists”.
5. View the results on the confirmation page. You may change specialists at any point during the game by repeating Steps 1-3 of this section.

### 4.1.4 More Administrator Options

The following steps should be followed to remove traders from the game or clear the entire database.

1. Login as “admin” if not already.
2. Click the “More Administrator Options” link in the “Administrator Options” box or at the bottom of any of the administrator pages.
3. Click the button on this page that corresponds to what you want to do.
4. If you chose “Clear database data”, you must now repeat the steps in section 4.1.1 to reconfigure the game.

### 4.1.5 Administrator: Ending the Game

1. The game will end automatically at the time you specified.
2. If you wish to end it early, click the “End Game” button in the “Administrator Option” box on the index page.
3. The trading screen will immediately be replaced by the results of the game, with traders

sorted by their net worth and the winning trader displayed in red at the top of the list.

## 4.2 All Traders

### 4.2.1 First Time Setup

1. Navigate to the main page and enter a username and password of your choice.
2. Click “Create New Account”.
3. Once you have created your account, you can log in with the username and password you created anytime you return to the site to play the game.

### 4.2.2 Playing the Game

1. Login with the username and password you created.
2. Click the “Trading Page” link in the “Login” box.
3. Your cash and stock holdings are shown in the “Portfolio” box.
4. To submit a buy or sell limit order
    a. Find the “Order” box on the trading screen.
    b. Select the security you want to buy or sell from the dropdown menu.
    c. Enter the price you want to sell at and the number of shares you want to sell.
    d. Click either the “Buy” or “Sell” button.
    e. Your order will appear shortly in your “Pending Transactions” box. Note that it may not appear if it is immediately completed.
    f. When all or a portion of your order is completed, the completed portion will appear in the “Completed Transactions” box.
5. To gain information about the true price of a security:
    a. Select the bid/ask graph corresponding to the security by selection it from the dropdown menu in the “Market Information” box.
    b. Read recent news releases relating to the security in the “News” box.

### 4.2.3 Specialists

Specialists should follow all the normal trading guidelines in Section 4.2.2 in addition to the following.

1. To view the limit order book for your security:
    a. Click the “Click Here to View Limit Order Book” link at the bottom left of your trading screen.
    b. A popup box will appear that shows all pending transactions for all the securities that you are the specialist for.
    c. Click the “close” link to close this box and resume trading.

### 4.2.4 End of Game

At the end of the game, the trading screen will immediately be replaced by the results of the game, with traders sorted by their net worth and the winning trader displayed in red at the top of the list.

## Section 5. Societal Issues

The societal concerns of the SCU Stock Market Simulation Model are negligible. The simulation is more environmentally friendly than the NYSE in the sense that there is no need for a physical stock exchange. All the orders and transactions are recorded electronically, and trading can be done from any computer with Internet access (thus eliminating the need for travel to a specific location for trading).

Another concern for appraisal was that students might become overly confident in their ability to manage stock market trading through the use of the simulation. This overconfidence might lead to disastrous results if a student attempted to manage his stock market portfolio with little knowledge or experience beyond the simulation. However, we determined that in an undergraduate classroom setting at Santa Clara University this risk was negligible. Certainly most students know there is a very real difference between playing a simulation with imaginary funds and risking your own savings.

There is a real benefit to society, however, that stems from the SCU Stock Market Simulation Model. This benefit is the experience that students can gain trading stocks just as they would on

the NYSE. They can learn valuable analytical skills without financial risk to themselves or others.

## Section 6. Conclusion

The SCU Stock Market Simulation Model has been completed in a manner that satisfies all essential requirements of our customer, Professor Sanjiv Das of the SCU Finance Department. Further upgrades are possible, but we have built a fully functional simulation model ready for use in a classroom. This simulation model is a valuable tool for all those who wish to study stock market behavior or to learn stock trading techniques without financial cost. In the short term, we expect that this simulation will be successful in giving SCU finance students a hands-on experience of market trading. Longer term, this project can be used in a variety of contexts: for example, it can be used to host a competition within the business school. The practical experience and the competitive nature of this simulation mean that many other applications are possible.

We gained significant experience in developing the SCU Stock Market Simulation Model. Specifically, we learned the use of new technologies in AJAX, MySQL, and PHP. It was also a very practical project, since we had the chance to develop a product for a real-world customer (Professor Sanjiv Das) while at the same time reporting to our faculty advisor (Professor Darren Atkinson). Finally, we had the chance to present our project to industry judges and others at the Senior Design Conference in the spring quarter. We are honored to have received the “Best in Session” and “Best in Conference” awards from the Conference. Our primary goal for this project, however, was to realize all essential requirements and to deliver a robust attractive system to our customer. In achieving this goal our project became a success.

# Appendix

## Stock Price Generation and News Release Algorithms

The following are the algorithm descriptions provided by the customer. They were implemented in PHP and used to generate both the “true” stock price and the news release stock price (which was within an administrator-specified percentage of the “true” stock price).

# Stock Process in the Trading Game

Sanjiv R. Das

April 13, 2006

## 1 Basic Set Up

Information disseminated in the trading model will depend on an underlying stochastic process followed by the stock price. This may be represented in discrete time as a tree that enumerates all the possible scenarios of stock prices.

It is in fact easier to represent this without a tree, especially from the viewpoint of the software engineer. The subroutine required would comprise 2 basic elements:

1. Generate the next stock price for the subsequent epoch on the tree.
2. Generate information about this stock price to be disseminated to the traders.

We present some notation to represent the above. We assume that the game will be played over a trading horizon $T$ (specified in minutes). We also specify the number of news releases we wish to have during the game, denoted $n$. Hence, the news release interval is specified as $h = T/n$. We index the time periods by variable $t$, such that $t = 1, 2, ..., n$.

We define the initial stock price on the tree $S_0$. The game administrator will specify also the growth rate of the stock ($\mu$) and the standard deviation of the growth rate ($\sigma$) as expected over the life of the game. For example, we may set $\mu = 0.10$, i.e. 10% and $\sigma = 0.30$, i.e. 30%.

Finally, we specify (a) the fraction of traders that will receive an information signal each period, denoted $m$, and (b) the information error $\eta$, expressed as a



**Figure 2.5 Stock Price Generation (1 of 2)**

standard deviation. For example, the fraction of traders may be $m = 0.55$, i.e. 55%, and the information error may be $\eta = 0.40$, i.e. 40%.

## 2 Generating the stock price path

Given a stock price at $t$, the next stock price is generated using the following equation:

$$S_{t+1} = S_t \exp\left[\left(\mu - \frac{1}{2}\sigma^2\right)h + \sigma\epsilon\sqrt{h}\right]$$

where $\epsilon \sim N(0,1)$, i.e. it is a standard random normal variable. This equation is the classic one used in finance for modeling stock price movements in discrete time as approximations of geometric Brownian motions.

Next, given $S_{t+1}$, we generate the information for the fraction of traders that are to receive partial information about the stock price. Suppose this number of traders is $L$. Then for each $i = 1, ..., L$, we do the following:

- Generate a random variable $x_i \sim N(0,1)$.
- Compute $y_i = S_{t+1} \times e^x$.
- If $y_i < S_{t+1}$, then the trader receives the following information:
  "Analysts expect the stock price to be worth at least $y_i$".

  If $y_i \geq S_{t+1}$, then the trader receives the following information:
  "Analysts expect the stock price to be worth at most $y_i$".

  We can of course tack on some verbiage about the market conditions and so on within which we embed this statement.

Repeat this for all $i$ (over all periods $t$).

Once the information has been broadcast then we allow the traders to trade over time interval $(t+1)$ which is of length $h$ minutes, before redoing the next stock price.

Hence, as is apparent, this is a simpler implementation of the "true" stock price process over the life of the trading game, and does not need a tree. The role of the traders as a collective group will be to use their partial bits of information and trade, thus sharing information through the process of trading, resulting in eventual price discovery. This exemplifies how markets really work.



**Figure 2.6 Stock Price Generation (2 of 2)**

## Works Cited

Das, Sanjiv. Guest Lecture. Software Engineering class. “Equity Markets Trading Game.” Spring 2006.

Joe Mocerino. "Ajax, the Way of the Future." Zunch. 2 Feb. 2006. <http://images.google.com/imgres?imgurl=http://blog.zunch.com/UserFiles/Image/ajaxDiagram.png&imgrefurl=http://blog.zunch.com/archive/2006/2/&h=879&w=698&sz=78&hl=en&start=1&um=1&tbnid=ndt8UhHIV2ekJM:&tbnh=146&tbnw=116&prev=/images%3Fq%3Dajax%2Bdiagram%26svnum%3D10%26um%3D1%26hl%3Den>